Manipulation and Bribery

AGT-MIRI

Spring 2026

- To represent an election with m candidates and n voters, we have to provide a description of the votes of the voters.

- To represent an election with m candidates and n voters, we have to provide a description of the votes of the voters.
- A vote is a permutation of the set of candidates.

- To represent an election with m candidates and n voters, we have to provide a description of the votes of the voters.
- A vote is a permutation of the set of candidates.
- Extensive: A vote for each player
  Which has size $O(mn)$.

- To represent an election with m candidates and n voters, we have to provide a description of the votes of the voters.
- A vote is a permutation of the set of candidates.
- Extensive: A vote for each player
  Which has size $O(mn)$.
- Weighted: A weighted set of votes.
  The weight indicates the number of players selecting this particular vote. The sum of the weights is the number of players.
  Which has size $O(m! \log n)$

- To represent an election with m candidates and n voters, we have to provide a description of the votes of the voters.
- A vote is a permutation of the set of candidates.
- Extensive: A vote for each player
  Which has size $O(mn)$.
- Weighted: A weighted set of votes.
  The weight indicates the number of players selecting this particular vote. The sum of the weights is the number of players.
  Which has size $O(m! \log n)$
- Prized: In addition to votes, a prize for each voter is given.
  This has additional size $O(n \log P)$ where P is the largest prize.

Coalition manipulation: scoring protocols

- We analyze the coalition manipulation problem on scoring protocols.

Coalition manipulation: scoring protocols

- We analyze the coalition manipulation problem on scoring protocols.
- A scoring protocol is defined bay a tuple $\alpha = (\alpha_1, \ldots, \alpha_m)$ with $\alpha_1 \geq \cdots \geq \alpha_m$.
- α_i is the number of points assigned to the i-th alternative in a vote.

Coalition manipulation: scoring protocols

- We analyze the coalition manipulation problem on scoring protocols.
- A scoring protocol is defined bay a tuple $\alpha = (\alpha_1, \ldots, \alpha_m)$ with $\alpha_1 \geq \cdots \geq \alpha_m$.
- α_i is the number of points assigned to the i-th alternative in a vote.
- In such kind of elections there is always a winner but there might be multiple winners.

Coalition manipulation: scoring protocols

- We analyze the coalition manipulation problem on scoring protocols.
- A scoring protocol is defined bay a tuple $\alpha = (\alpha_1, \ldots, \alpha_m)$ with $\alpha_1 \geq \cdots \geq \alpha_m$.
- α_i is the number of points assigned to the i-th alternative in a vote.
- In such kind of elections there is always a winner but there might be multiple winners.
- We assume that the input is given in weighted form.

Manipulation MP_α

Given: A set S of weighted votes over m candidates, the weights for a set T of votes, and a preferred candidate p from among the m candidates.

Question: Is there a way to cast the votes in T such that p wins the election with respect to α?

Manipulation MP_α

Given: A set S of weighted votes over m candidates, the weights for a set T of votes, and a preferred candidate p from among the m candidates.

Question: Is there a way to cast the votes in T such that p wins the election with respect to α?

Unique Manipulation UMP_α

Given: A set S of weighted votes over m candidates, the weights for a set T of voters, and a preferred candidate p from among the m candidates.

Question: Is there a way to cast the votes in T such that p is the unique winner of the election with respect to α?

A complexity dichotomy

A complexity dichotomy

Theorem

For each m and each scoring protocol $\alpha = (\alpha_1, \ldots, \alpha_m)$, MP_α and UMP_α are in P if $\alpha_2 = \cdots = \alpha_m$, and are NP-complete in all other cases.

Proof of theorem: Polynomial cases

- If $\alpha_1 = \cdots = \alpha_m$, all candidates are always tied, so all of them are winners and unique winners only if $m = 1$.

Proof of theorem: Polynomial cases

- If $\alpha_1 = \cdots = \alpha_m$, all candidates are always tied, so all of them are winners and unique winners only if $m = 1$.
- If $\alpha_1 > \alpha_2 = \cdots = \alpha_m$, by the previous claim, the election is eqivalent to $(1, 0, \ldots, 0)$.

Proof of theorem: Polynomial cases

- If $\alpha_1 = \cdots = \alpha_m$, all candidates are always tied, so all of them are winners and unique winners only if $m = 1$.
- If $\alpha_1 > \alpha_2 = \cdots = \alpha_m$, by the previous claim, the election is eqivalent to $(1, 0, \ldots, 0)$.
- But, we know that plurality is manipulable in polynomial time.

Proof of theorem: hard cases

- By the previous claim we can assume that $\alpha_m = 0$.

Proof of theorem: hard cases

- By the previous claim we can assume that $\alpha_m = 0$.
- The reduction is from
  Partition
  Given: $n \geq 1$ positive integers $k_1, \ldots, k_n$ that sum to $2K$.
  Question: Is there a subset of the integers that sums to K?

Proof of theorem: hard cases

- By the previous claim we can assume that $\alpha_m = 0$.
- The reduction is from
  Partition
  Given: $n \geq 1$ positive integers $k_1, \ldots, k_n$ that sum to $2K$.
  Question: Is there a subset of the integers that sums to K?
- Let $\ell = |\{i \mid 1 \leq i \leq m, \alpha_i = 0\}| - 1$.

Proof of theorem: hard cases

- By the previous claim we can assume that $\alpha_m = 0$.
- The reduction is from
  Partition
  Given: $n \geq 1$ positive integers $k_1, \ldots, k_n$ that sum to $2K$.
  Question: Is there a subset of the integers that sums to K?
- Let $\ell = |\{i \mid 1 \leq i \leq m, \alpha_i = 0\}| - 1$.
- Since $\alpha_m = 0$, $\ell \geq 0$.

Proof of theorem: hard cases

- By the previous claim we can assume that $\alpha_m = 0$.
- The reduction is from
  Partition
  Given: $n \geq 1$ positive integers $k_1, \ldots, k_n$ that sum to $2K$.
  Question: Is there a subset of the integers that sums to K?
- Let $\ell = |\{i \mid 1 \leq i \leq m, \alpha_i = 0\}| - 1$.
- Since $\alpha_m = 0$, $\ell \geq 0$.
- The proof requires a construction for $\ell = 1$ and another for $\ell \neq 1$.
  We present only the case $\ell \neq 1$.

Proof of theorem

- Let $\ell = |\{i \mid 1 \leq i \leq m, \alpha_i = 0\}| - 1$.
- Let $r = m - \ell - 3$.
- As $\alpha_2 \neq 0$, $r \geq 0$.

Proof of theorem

- Let $\ell = |\{i \mid 1 \leq i \leq m, \alpha_i = 0\}| - 1$.
- Let $r = m - \ell - 3$.
- As $\alpha_2 \neq 0$, $r \geq 0$.
- The election has m candidates: $p, a, b, c_1, \ldots, c_\ell, d_1, \ldots, d_r$.

Proof of theorem

- Let $\ell = |\{i \mid 1 \leq i \leq m, \alpha_i = 0\}| - 1$.
- Let $r = m - \ell - 3$.
- As $\alpha_2 \neq 0$, $r \geq 0$.
- The election has m candidates: $p, a, b, c_1, \ldots, c_\ell, d_1, \ldots, d_r$.
- The set of voters S is created in two parts, $S = S_1 \cup S_2$.

The case $\ell \neq 1$: S_1

The case $\ell \neq 1$: S_1

- One vote of weight $2K(2\alpha_1 - \alpha_{r+2}) - 1$
  $a > d_1 > \cdots > d_r > b > p > c_1 > \cdots > c_\ell$.

The case $\ell \neq 1$: S_1

- One vote of weight $2K(2\alpha_1 - \alpha_{r+2}) - 1$
  $a > d_1 > \cdots > d_r > b > p > c_1 > \cdots > c_\ell$.
- One vote of weight $2K(2\alpha_1 - \alpha_{r+2}) - 1$
  $b > d_1 > \cdots > d_r > a > p > c_1 > \cdots > c_\ell$.

The case $\ell \neq 1$: S_1

- One vote of weight $2K(2\alpha_1 - \alpha_{r+2}) - 1$
  $a > d_1 > \cdots > d_r > b > p > c_1 > \cdots > c_\ell$.
- One vote of weight $2K(2\alpha_1 - \alpha_{r+2}) - 1$
  $b > d_1 > \cdots > d_r > a > p > c_1 > \cdots > c_\ell$.
- For $1 \leq i \leq \ell$, one vote of weight $4\alpha_1 K - 1$
  $c_i > d_1 > \cdots > d_r > c_{1+(i \mod \ell)} > a > b > p > \ldots$.

  "..." means that the remaining candidates follow in some arbitrary order.

The case $\ell \neq 1$: S_1

- $score_{S_1}(p) = 0$

The case $\ell \neq 1$: S_1

- $score_{S_1}(p) = 0$
- $score_{S_1}(a) = score_{S_1}(b) = 2K(2\alpha_1 - \alpha_{r+2} - 1)(\alpha_1 + \alpha_{r+2})$

The case $\ell \neq 1$: S_1

- $score_{S_1}(p) = 0$
- $score_{S_1}(a) = score_{S_1}(b) = 2K(2\alpha_1 - \alpha_{r+2} - 1)(\alpha_1 + \alpha_{r+2})$
- $score_{S_1}(d_i) \leq score_{S_1}(d_1)$

The case $\ell \neq 1$: S_1

- $score_{S_1}(p) = 0$
- $score_{S_1}(a) = score_{S_1}(b) = 2K(2\alpha_1 - \alpha_{r+2} - 1)(\alpha_1 + \alpha_{r+2})$
- $score_{S_1}(d_i) \leq score_{S_1}(d_1)$
- $score_{S_1}(d_1) = 2(2K(2\alpha_1 - \alpha_{r+2} - 1) + \ell(4\alpha_1 K - 1))\alpha_2$

The case $\ell \neq 1$: S_2

- We use the construction of the first Claim to construct S_2 taking $s = score_{S_1}(d_1) + 1$.

The case $\ell \neq 1$: S_2

- We use the construction of the first Claim to construct S_2 taking $s = score_{S_1}(d_1) + 1$.
- We know:
  - $score_{S_2}(p) = score_{S_2}(x)$, for $x \in \{a, b, c_1, \ldots, c_\ell\}$, and
  - $score_{S_2}(d_i) + score_{S_1}(d_1) + 1 \leq score_{S_2}(p)$, for $1 \leq i \leq r$.

The case $\ell \neq 1$: S_2

- We use the construction of the first Claim to construct S_2 taking $s = score_{S_1}(d_1) + 1$.
- We know:
  - $score_{S_2}(p) = score_{S_2}(x)$, for $x \in \{a, b, c_1, \ldots, c_\ell\}$, and
  - $score_{S_2}(d_i) + score_{S_1}(d_1) + 1 \leq score_{S_2}(p)$, for $1 \leq i \leq r$.
  - Let $s_2 = score_{S_2}(p)$

The case $\ell \neq 1$: S_2

- We use the construction of the first Claim to construct S_2 taking $s = score_{S_1}(d_1) + 1$.
- We know:
  - $score_{S_2}(p) = score_{S_2}(x)$, for $x \in \{a, b, c_1, \ldots, c_\ell\}$, and
  - $score_{S_2}(d_i) + score_{S_1}(d_1) + 1 \leq score_{S_2}(p)$, for $1 \leq i \leq r$.
  - Let $s_2 = score_{S_2}(p)$
- Let $S = S_1 \cup S_2$, and set the weights of the n voters in T to $2k_1(\alpha_1 - \alpha_{r+2}), 2k_2(\alpha_1 - \alpha_{r+2}), \ldots, 2k_n(\alpha_1 - \alpha_{r+2})$.
- Let us show that this is a reduction.

The case $\ell \neq 1$: Partition has solution

- Let $I \subseteq [n]$ be such that $\sum_{i \in I} k_i = K$.

The case $\ell \neq 1$: Partition has solution

- Let $I \subseteq [n]$ be such that $\sum_{i \in I} k_i = K$.
- The vote of the i-th voter in T is

  $p > d_1 > \cdots > d_r > a > b > c_1 > \cdots > c_\ell$, if $i \in I$

  $p > d_1 > \cdots > d_r > b > a > c_1 > \cdots > c_\ell$, if $i \notin I$

The case $\ell \neq 1$: Partition has solution

- Let $I \subseteq [n]$ be such that $\sum_{i \in I} k_i = K$.
- The vote of the i-th voter in T is
  $p > d_1 > \cdots > d_r > a > b > c_1 > \cdots > c_\ell$, if $i \in I$
  $p > d_1 > \cdots > d_r > b > a > c_1 > \cdots > c_\ell$, if $i \notin I$
- This is equivalent to say that we have a vote of each type with weight $2K(\alpha_1 + \alpha_{r+2})$.

The case $\ell \neq 1$: Partition has solution

The case $\ell \neq 1$: Partition has solution

- $score(p) = s_2 + 4\alpha_1 K(\alpha_1 + \alpha_{r+2})$
- $score(a) = score(b) = (2K(2\alpha_1 - \alpha_{r+2} - 1)(\alpha_1 + \alpha_{r+2}) + s_2 + 2K(\alpha_1 + \alpha_{r+2})\alpha_{r+2} = s_2 + (4K\alpha_1 - 1)(\alpha_1 + \alpha_{r+2}) < score(p)$

The case $\ell \neq 1$: Partition has solution

- $score(p) = s_2 + 4\alpha_1 K(\alpha_1 + \alpha_{r+2})$
- $score(a) = score(b) = (2K(2\alpha_1 - \alpha_{r+2} - 1)(\alpha_1 + \alpha_{r+2}) + s_2 + 2K(\alpha_1 + \alpha_{r+2})\alpha_{r+2} = s_2 + (4K\alpha_1 - 1)(\alpha_1 + \alpha_{r+2}) < score(p)$
- For $1 \leq i \leq \ell$, $score(c_i) = (4\alpha_1 K - 1)(\alpha_1 + \alpha_{r+2}) + s_2 < score(p)$

The case $\ell \neq 1$: Partition has solution

- $score(p) = s_2 + 4\alpha_1 K(\alpha_1 + \alpha_{r+2})$
- $score(a) = score(b) = (2K(2\alpha_1 - \alpha_{r+2} - 1)(\alpha_1 + \alpha_{r+2}) + s_2 + 2K(\alpha_1 + \alpha_{r+2})\alpha_{r+2} = s_2 + (4K\alpha_1 - 1)(\alpha_1 + \alpha_{r+2}) < score(p)$
- For $1 \leq i \leq \ell$, $score(c_i) = (4\alpha_1 K - 1)(\alpha_1 + \alpha_{r+2}) + s_2 < score(p)$
- For $1 \leq i \leq r$, $score(d_i) \leq score(d_1) = score_S(d_1) + score_T(d_1) < score_S(p) + score_T(p) = score(p)$

The case $\ell \neq 1$: Votes in T can be cast

- Suppose the votes in T are cast in such a way that p is a winner of the election.
- Without loss of generality, assume that p is the most preferred candidate in the preference order of each voter in T. Then $score(p) = s_2 + 4\alpha_1 K(\alpha_1 + \alpha_{r+2})$.

The case $\ell \neq 1$: Votes in T can be cast

- Suppose that for some i, $score_T(c_i) > 0$.

The case $\ell \neq 1$: Votes in T can be cast

- Suppose that for some i, $score_T(c_i) > 0$.
- Since all weights of T are multiples of $2(\alpha_1 + \alpha_{r+2})$, $score_T(c_i) \geq 2(\alpha_1 + \alpha_{r+2})$.

The case $\ell \neq 1$: Votes in T can be cast

- Suppose that for some i, $score_T(c_i) > 0$.
- Since all weights of T are multiples of $2(\alpha_1 + \alpha_{r+2})$, $score_T(c_i) \geq 2(\alpha_1 + \alpha_{r+2})$.
- But then, $score(c_i) \geq (4\alpha_1 K - 1)(\alpha_1 + \alpha_{r+2}) + s_2 + 2(\alpha_1 + \alpha_{r+2}) = s_2 + (4\alpha_1 K - 1)(\alpha_1 + \alpha_{r+2}) > score(p)$.

The case $\ell \neq 1$: Votes in T can be cast

- Suppose that for some i, $score_T(c_i) > 0$.
- Since all weights of T are multiples of $2(\alpha_1 + \alpha_{r+2})$, $score_T(c_i) \geq 2(\alpha_1 + \alpha_{r+2})$.
- But then, $score(c_i) \geq (4\alpha_1 K - 1)(\alpha_1 + \alpha_{r+2}) + s_2 + 2(\alpha_1 + \alpha_{r+2}) = s_2 + (4\alpha_1 K - 1)(\alpha_1 + \alpha_{r+2}) > score(p)$.
- This contradicts the assumption that p is a winner.

The case $\ell \neq 1$: Votes in T can be cast

- There are only $\ell + 1$ 0's in α. Since $score_T(c_i) = 0$, for $1 \leq i \leq \ell$, each voter in T must have at least one of a or b somewhere between (inclusively) 1st and $(r + 2)$nd in his/her preference order.

The case $\ell \neq 1$: Votes in T can be cast

- There are only $\ell + 1$ 0's in α. Since $score_T(c_i) = 0$, for $1 \leq i \leq \ell$, each voter in T must have at least one of a or b somewhere between (inclusively) 1st and $(r+2)$nd in his/her preference order.
- So, $score_T(a) + score_T(b) \geq 4K(\alpha_1 + \alpha_{r+2})\alpha_{r+2}$.

The case $\ell \neq 1$: Votes in T can be cast

- Suppose that $score_T(a) > 2K(\alpha_1 + \alpha_{r+2})\alpha_{r+2}$.

The case $\ell \neq 1$: Votes in T can be cast

- Suppose that $score_T(a) > 2K(\alpha_1 + \alpha_{r+2})\alpha_{r+2}$.
- Since all weights of T are multiples of $2(\alpha_1 + \alpha_{r+2})$, it follows that $score_T(a) \geq 2K(\alpha_1 + \alpha_{r+2})\alpha_{r+2} + 2(\alpha_1 + \alpha_{r+2}) = (2K\alpha_{r+2} + 2)(\alpha_1 + \alpha_{r+2})$.

The case $\ell \neq 1$: Votes in T can be cast

- Suppose that $score_T(a) > 2K(\alpha_1 + \alpha_{r+2})\alpha_{r+2}$.
- Since all weights of T are multiples of $2(\alpha_1 + \alpha_{r+2})$, it follows that $score_T(a) \geq 2K(\alpha_1 + \alpha_{r+2})\alpha_{r+2} + 2(\alpha_1 + \alpha_{r+2}) = (2K\alpha_{r+2} + 2)(\alpha_1 + \alpha_{r+2})$.
- Then (keeping in mind our choice of S_2)
  $score(a) \geq 2K(\alpha_1+\alpha_{r+2})(\alpha_1+\alpha_{r+2})+s_2+(2K\alpha_{r+2}+2)(\alpha_1+\alpha_{r+2}) =$
  $s2 + (2K(2\alpha_1 - \alpha_{r+2}) + 1 + 2K\alpha_{r+2})(\alpha_1 + \alpha_{r+2}) =$
  $s_2 + (4K\alpha_1 + 1)(\alpha_1 + \alpha_{r+2}) > score(p)$.

The case $\ell \neq 1$: Votes in T can be cast

- Suppose that $score_T(a) > 2K(\alpha_1 + \alpha_{r+2})\alpha_{r+2}$.
- Since all weights of T are multiples of $2(\alpha_1 + \alpha_{r+2})$, it follows that $score_T(a) \geq 2K(\alpha_1 + \alpha_{r+2})\alpha_{r+2} + 2(\alpha_1 + \alpha_{r+2}) = (2K\alpha_{r+2} + 2)(\alpha_1 + \alpha_{r+2})$.
- Then (keeping in mind our choice of S_2) $score(a) \geq 2K(\alpha_1+\alpha_{r+2})(\alpha_1+\alpha_{r+2})+s_2+(2K\alpha_{r+2}+2)(\alpha_1+\alpha_{r+2}) = s2 + (2K(2\alpha_1 - \alpha_{r+2}) + 1 + 2K\alpha_{r+2})(\alpha_1 + \alpha_{r+2}) = s_2 + (4K\alpha_1 + 1)(\alpha_1 + \alpha_{r+2}) > score(p)$.
- Contradicting the assumption that p is a winner.

The case $\ell \neq 1$: Votes in T can be cast

- Since a and b are completely symmetric, it follows that $score_T(a) = score_T(b) = 2K(\alpha_1 + \alpha_{r+2})\alpha_{r+2}$.

The case $\ell \neq 1$: Votes in T can be cast

- Since a and b are completely symmetric, it follows that $score_T(a) = score_T(b) = 2K(\alpha_1 + \alpha_{r+2})\alpha_{r+2}$.
- But then the weights of those voters in T who prefer a to b sum to $2K(\alpha_1 + \alpha_{r+2})$.

The case $\ell \neq 1$: Votes in T can be cast

- Since a and b are completely symmetric, it follows that $score_T(a) = score_T(b) = 2K(\alpha_1 + \alpha_{r+2})\alpha_{r+2}$.
- But then the weights of those voters in T who prefer a to b sum to $2K(\alpha_1 + \alpha_{r+2})$.
- But then, there is a subset $I \subseteq [n]$ such that $\sum_{i \in I} 2k_i(\alpha_1 + \alpha_{r+2}) = 2K(\alpha_1 + \alpha_{r+2})$.

The case $\ell \neq 1$: Votes in T can be cast

- Since a and b are completely symmetric, it follows that $score_T(a) = score_T(b) = 2K(\alpha_1 + \alpha_{r+2})\alpha_{r+2}$.
- But then the weights of those voters in T who prefer a to b sum to $2K(\alpha_1 + \alpha_{r+2})$.
- But then, there is a subset $I \subseteq [n]$ such that $\sum_{i \in I} 2k_i(\alpha_1 + \alpha_{r+2}) = 2K(\alpha_1 + \alpha_{r+2})$.
- So, $\sum_{i \in I} k_i = K$.

Bribery Problem

- We will study the problem on variants of plurality.

Bribery Problem

- We will study the problem on variants of plurality.
- Plurality with weights:

Bribery Problem

- We will study the problem on variants of plurality.
- Plurality with weights:
  - Voter i has weight w_i.

Bribery Problem

- We will study the problem on variants of plurality.
- Plurality with weights:
  - Voter i has weight w_i.
  - Voter i gives w_i points to its most preferred candidate and 0 to the others.

Bribery Problem

- We will study the problem on variants of plurality.
- Plurality with weights:
  - Voter i has weight w_i.
  - Voter i gives w_i points to its most preferred candidate and 0 to the others.
  - The candidate with the higher number of votes wins.

Bribery with money

Bribery with money

Name: Plurality-weighted-$bribery

Input: *A set C of m candidates. A collection V of n voters specified via: a preference profile $\succ$, weights $(w_1, \ldots, w_n)$, and their prices $(p_1, \ldots, p_n)$. A distinguished candidate $c \in C$ and a non-negative integer k, the budget.*

Question: *Is there a set B of voters such that $\sum_{i \in B} p_i \leq k$ and there is a way to bribe the voters from B in such a way that c becomes a winner?*

Plurality

Theorem

Plurality-bribery belongs to P

Plurality

Theorem

Plurality-bribery belongs to P

Proof.

Plurality

Theorem

Plurality-bribery belongs to P

Proof.

Consider the following algorithm.

Plurality

Theorem

Plurality-bribery belongs to P

Proof.

Consider the following algorithm.

- Initially we have bribed zero voters. We check whether c is a winner on $\succ$. If so, we answer yes.

Plurality

Theorem

Plurality-bribery belongs to P

Proof.

Consider the following algorithm.

- Initially we have bribed zero voters. We check whether c is a winner on $\succ$. If so, we answer yes.
- Otherwise, until doing so will exceed the bribe limit, pick any current winner b, bribe one of the voters ranking first b to rank first c. If c is now a winner answer yes.

Plurality

Theorem

Plurality-bribery belongs to P

Proof.

Consider the following algorithm.

- Initially we have bribed zero voters. We check whether c is a winner on $\succ$. If so, we answer yes.
- Otherwise, until doing so will exceed the bribe limit, pick any current winner b, bribe one of the voters ranking first b to rank first c. If c is now a winner answer yes.
- Answer no.

Plurality

Proof (cont).

Plurality

Proof (cont).

- If the algorithm says yes, obviously bribery is possible.

Plurality

Proof (cont).

- If the algorithm says yes, obviously bribery is possible.
- An easy induction proof shows that, if it is possible to ensure that c is a winner via at most k bribes, our algorithm answers yes

Plurality with weights

Theorem

Plurality-weighted-$bribery is NP-complete, even for two candidates

Plurality with weights

Theorem

Plurality-weighted-$bribery is NP-complete, even for two candidates

Proof.

Plurality with weights

Theorem

Plurality-weighted-$bribery is NP-complete, even for two candidates

Proof.

We construct a reduction from PARTITION:
Given integers $x_1, \ldots, x_n$ with $\sum_{i=1}^{n} x_i = 2x$.
Is there a set $S \subseteq \{1, \ldots n\}$ so that $\sum_{i \in S} x_i = \sum_{j \notin S} x_j = x$?

Plurality with weights

Theorem

Plurality-weighted-$bribery is NP-complete, even for two candidates

Proof.

We construct a reduction from PARTITION:
Given integers $x_1, \ldots, x_n$ with $\sum_{i=1}^{n} x_i = 2x$.
Is there a set $S \subseteq \{1, \ldots n\}$ so that $\sum_{i \in S} x_i = \sum_{j \notin S} x_j = x$?

- The election will have two candidates, a and c, and n voters.

Plurality with weights

Theorem

Plurality-weighted-$bribery is NP-complete, even for two candidates

Proof.

We construct a reduction from PARTITION:
Given integers $x_1, \ldots, x_n$ with $\sum_{i=1}^{n} x_i = 2x$.
Is there a set $S \subseteq \{1, \ldots n\}$ so that $\sum_{i \in S} x_i = \sum_{j \notin S} x_j = x$?

- The election will have two candidates, a and c, and n voters.
- Voter i has weight and prize equal to s_i.

Plurality with weights

Theorem

Plurality-weighted-$bribery is NP-complete, even for two candidates

Proof.

We construct a reduction from PARTITION:
Given integers $x_1, \ldots, x_n$ with $\sum_{i=1}^{n} x_i = 2x$.
Is there a set $S \subseteq \{1, \ldots n\}$ so that $\sum_{i \in S} x_i = \sum_{j \notin S} x_j = x$?

- The election will have two candidates, a and c, and n voters.
- Voter i has weight and prize equal to s_i.
- Every voter prefers a to c.

Plurality with weights

Theorem

Plurality-weighted-$bribery is NP-complete, even for two candidates

Proof.

We construct a reduction from PARTITION:
Given integers $x_1, \ldots, x_n$ with $\sum_{i=1}^{n} x_i = 2x$.
Is there a set $S \subseteq \{1, \ldots n\}$ so that $\sum_{i \in S} x_i = \sum_{j \notin S} x_j = x$?

- The election will have two candidates, a and c, and n voters.
- Voter i has weight and prize equal to s_i.
- Every voter prefers a to c.
- $k = x$.

Plurality

Proof (cont).

Plurality

Proof (cont).

- If the partition instance has a solution S, we can bribe the voters in S. We expend al the budget and make c a winner.

Plurality

Proof (cont).

- If the partition instance has a solution S, we can bribe the voters in S. We expend al the budget and make c a winner.
- Otherwise, for any set S with cost $\leq x$, S asigns $\leq x$ points to c, but $V \setminus S$ assigns $> x$ points to a.

Plurality

Proof (cont).

- If the partition instance has a solution S, we can bribe the voters in S. We expend al the budget and make c a winner.
- Otherwise, for any set S with cost $\leq x$, S asigns $\leq x$ points to c, but $V \setminus S$ assigns $> x$ points to a.
  Therefore, the bribery problem has no solution.

Plurality with weights

Theorem

Both Plurality-$bribery and Plurality-weighted bribery are in P.

Plurality with weights

Theorem

Both Plurality-$bribery and Plurality-weighted bribery are in P.

Proof.

Plurality with weights

Theorem

Both Plurality-$bribery and Plurality-weighted bribery are in P.

Proof.

- Assume that c will have r votes after the bribery (or in the weighted case, vote weight r), where r is some number to be specified later.

Plurality with weights

Theorem

Both Plurality-$bribery and Plurality-weighted bribery are in P.

Proof.

- Assume that c will have r votes after the bribery (or in the weighted case, vote weight r), where r is some number to be specified later.
- To make c a winner, we need to make sure that everyone else gets at most r votes.

Plurality with weights

Plurality with weights

Proof (cont).

Plurality with weights

Proof (cont).

- We have to choose enough cheapest (heaviest) voters of candidates that defeat c so that after bribing them to vote for c each candidate other than c has at most r votes.

Plurality with weights

Proof (cont).

- We have to choose enough cheapest (heaviest) voters of candidates that defeat c so that after bribing them to vote for c each candidate other than c has at most r votes.
- We have to make sure that c gets at least r votes by bribing the cheapest (the heaviest) of the remaining voters.

Plurality with weights

Proof (cont).

- We have to choose enough cheapest (heaviest) voters of candidates that defeat c so that after bribing them to vote for c each candidate other than c has at most r votes.
- We have to make sure that c gets at least r votes by bribing the cheapest (the heaviest) of the remaining voters.
- If during this process c ever becomes a winner, without exceeding the budget, then we know that bribery is possible.

Plurality with weights

Plurality with weights

Proof (cont).

Plurality with weights

Proof (cont).

- How do we pick the value of r?

Plurality with weights

Proof (cont).

- How do we pick the value of r?
- In the case of plurality-$bribery, we can simply run the above procedure for all possible values of r, i.e., $0 \leq r \leq n$, and accept exactly if it succeeds for at least one of them.

Plurality with weights

Proof (cont).

- How do we pick the value of r?
- In the case of plurality-$bribery, we can simply run the above procedure for all possible values of r, i.e., $0 \leq r \leq n$, and accept exactly if it succeeds for at least one of them.
- For plurality-weighted-bribery it is enough to try all values r that can be obtained as a vote weight of some candidate (other than c) via bribing some number of his or her heaviest voters.

Plurality with weights

Proof (cont).

- How do we pick the value of r?
- In the case of plurality-$bribery, we can simply run the above procedure for all possible values of r, i.e., $0 \leq r \leq n$, and accept exactly if it succeeds for at least one of them.
- For plurality-weighted-bribery it is enough to try all values r that can be obtained as a vote weight of some candidate (other than c) via bribing some number of his or her heaviest voters.
- There are only polynomially many such values and so the whole algorithm works in polynomial time.

Negative bribery

Negative bribery

- In the previous algorithms voters are bribed to vote for the desired candidate.

Negative bribery

- In the previous algorithms voters are bribed to vote for the desired candidate.
- This might made the bribery easily detectable.

Negative bribery

- In the previous algorithms voters are bribed to vote for the desired candidate.
- This might made the bribery easily detectable.
- To minimize this effect we would like to bribe voters to vote for other candidates instead of c.

Negative bribery

- In the previous algorithms voters are bribed to vote for the desired candidate.
- This might made the bribery easily detectable.
- To minimize this effect we would like to bribe voters to vote for other candidates instead of c.
- The negative-bribery version of a bribery problem is the same problem with the restriction that it is illegal to bribe people to vote for the designed candidate.

Negative bribery

Theorem

Plurality-negative-$bribery belongs to P.

Negative bribery

Theorem

Plurality-negative-$bribery belongs to P.

Proof.

- Let (C, V, c, k) be the bribery instance we want to solve.

Negative bribery

Theorem

Plurality-negative-$bribery belongs to P.

Proof.

- Let (C, V, c, k) be the bribery instance we want to solve.
- We need to make c a winner by taking votes away from popular candidates and distributing them among the less popular ones.

Negative bribery

Theorem

Plurality-negative-$bribery belongs to P.

Proof.

- Let (C, V, c, k) be the bribery instance we want to solve.
- We need to make c a winner by taking votes away from popular candidates and distributing them among the less popular ones.
- For a candidate a, define $Sc(a)$ to be the total vote weight of voters who most prefer a.

Negative bribery

Negative bribery

Proof (cont.)

Negative bribery

Proof (cont.)

- We partition the set of all candidates into three sets: candidates that

Negative bribery

Proof (cont.)

- We partition the set of all candidates into three sets: candidates that
  - defeat c, from whom votes need to be taken away
  - are defeated by c, to whom we can give extra votes, and
  - have the same score as p.

Negative bribery

Proof (cont.)

- We partition the set of all candidates into three sets: candidates that
  - defeat c, from whom votes need to be taken away
  - are defeated by c, to whom we can give extra votes, and
  - have the same score as p.

  and define

Negative bribery

Proof (cont.)

- We partition the set of all candidates into three sets: candidates that
  - defeat c, from whom votes need to be taken away
  - are defeated by c, to whom we can give extra votes, and
  - have the same score as p.

  and define

$$C_{above} = \{a | a \in C, Sc(a) > Sc(c)\}.$$
$$C_{below} = \{a | a \in C, Sc(a) < Sc(c)\}.$$
$$C_{equal} = \{a | a \in C, Sc(a) = Sc(c)\}.$$

Negative bribery

Proof (cont.)

Negative bribery

Proof (cont.)

- Voters have weight 1, so we

Negative bribery

Proof (cont.)

- Voters have weight 1, so we
  - will bribe no voters into or out of C_{equal} and
  - won't bribe voters to move within their own "group," e.g., bribing a voter to shift from one C_{below} candidate to another.

Negative bribery

Proof (cont.)

- Voters have weight 1, so we
  - will bribe no voters into or out of C_{equal} and
  - won't bribe voters to move within their own "group," e.g., bribing a voter to shift from one C_{below} candidate to another.
- To make sure that c becomes a winner, bribe, for $a \in C_{above}$, $Sc(a) - Sc(c)$ voters.

Negative bribery

Proof (cont.)

- Voters have weight 1, so we
  - will bribe no voters into or out of C_{equal} and
  - won't bribe voters to move within their own "group," e.g., bribing a voter to shift from one C_{below} candidate to another.
- To make sure that c becomes a winner, bribe, for $a \in C_{above}$, $Sc(a) - Sc(c)$ voters.
- The number of votes that a candidate $a \in C_{below}$ can accept without preventing c from winning is $Sc(c) - Sc(a)$.

Negative bribery

Proof (cont.)

- Voters have weight 1, so we
  - will bribe no voters into or out of C_{equal} and
  - won't bribe voters to move within their own "group," e.g., bribing a voter to shift from one C_{below} candidate to another.
- To make sure that c becomes a winner, bribe, for $a \in C_{above}$, $Sc(a) - Sc(c)$ voters.
- The number of votes that a candidate $a \in C_{below}$ can accept without preventing c from winning is $Sc(c) - Sc(a)$.
- Then a negative bribery is possible if

$$\sum_{a \in C_{above}} (Sc(a) - Sc(c)) \leq \sum_{a \in C_{below}} (Sc(c) - Sc(a)).$$

Negative bribery

Theorem

Plurality-weighted-negative-bribery is NP-complete

Negative bribery

Theorem

Plurality-weighted-negative-bribery is NP-complete

Proof.

We construct a reduction from Partition.

Negative bribery

Theorem

Plurality-weighted-negative-bribery is NP-complete

Proof.

We construct a reduction from Partition.

- Let $\{x_1, \ldots, x_n\}$ be a sequence of non-negative integers. Let $x_1 + \cdots + x_n = 2X$.
- The election has three candidates $c, a-1, a_2$ and the bribery budget is $k = n + 1$.
- There are $n + 1$ weighted voters:
  - v_0 with weight X, whose preferences are $s > a_1 > a_2$, and
  - $v_1, \ldots, v_n$ with weights $s_1, \ldots, s_n$, each with preferences $a1 > a2 > c$.

Negative bribery

Proof (cont.)

Negative bribery

Proof (cont.)

- The goal of the briber is to ensure c's victory via bribing up to $n+1$ voters (i.e., all)

Negative bribery

Proof (cont.)

- The goal of the briber is to ensure c's victory via bribing up to $n+1$ voters (i.e., all)
- The only reasonable bribe is to transfer the vote of v_i, $1 \leq i \leq n$, from a_1 to a_2.

Negative bribery

Proof (cont.)

- The goal of the briber is to ensure c's victory via bribing up to $n+1$ voters (i.e., all)
- The only reasonable bribe is to transfer the vote of v_i, $1 \leq i \leq n$, from a_1 to a_2.
- Then, A is a solution to partition iff bribing A makes c a winner.

Bribery: Approval voting

Bribery: Approval voting

Theorem

Approval-bribery is NP-complete

Recall that Approval-Manipulation can be solved in polynomial time.

References

- E. Hemaspaandra, L.A. Hemaspaandra. Dichotomy for voting systems. Journal of Computer and System Sciences 73(1):73–83, 2007
- P. Faliszewski, E. Hemaspaandra, L.A. Hemaspaandra. How Hard Is Bribery in Elections?. Journal of Artificial Intelligence Research 35:485-532, 2009
- F. Brandt et al., Eds. Handbook of Computational Choice Theory, Cambridge University Press, 2016.